propDB Documentation

NYU WIRELESS

July 6, 2015

1 Standard Use

In addition to providing direct access to propagation data through physical hard drives, NYU WIRELESS provides its affiliate access to "propDB", an online portal to an in-house database server which hosts raw and processed data from each measurement campaign performed by the organization.

Access is granted to board members and persons specifically suggested by board members. Each user gets a username / password pair that can be used to: log onto the site, allowing examination and search of the data; and download data to local workstations.

1.1 Logging On

You can reach the webpage by going to `http://data.nyuwireless.com`. On this page, you will be presented with a username / password gate. Enter your username and password to proceed.

Next you will be presented with an electronic version of the EULA that you signed in order to gain access to the data. At the bottom of the page, enter your name, email, title at the company you work for and confirm the current date. Finally, check the "I agree to the terms" checkbox.

Clicking submit constitutes your acceptance of the EULA agreement. Clicking submit saves your entry in several logs accessible to webmasters and proprietors.

1.2 Data Query

Users can cycle through available campaigns by clicking the [**Frequency** (·)] button at the left of the navigation bar. A different set of statistics has been computed for each campaign. Each of the available statistics can be used to search through the full dataset.

The set of boxes at the left side of the screen provides a means to do this — selecting any combination of query values and clicking the [**Filter**] button on the nav bar narrows down the presented dataset to only those raw measurements which correspond to the filter.

The set of all "valid" measurement populates the list at the right side of the screen. Clicking on any of the names in this list will bring up an image corresponding to the processed Power Delay Profile (PDP) for the selected raw measurement. If no processed image exists, a message "*We apologize, but the requested image cannot be loaded.*" is displayed instead.

Any single query to the dataset is guaranteed to yield a nonempty subset of the data; however, multiple queries at the same time do not [provide] the same guarantee.

1.3 Download

After having narrowed down the database to the subset you desire, you can download all the files in question by clicking the [**Download Queries Files** (·)] button. This will bring you to a new page that contains information regarding your request.

We provide data files over `sftp` connection. The server's hostname is `data.nyuwireless.com` and your login username is the username for the `sftp` service. Be sure to note the specified `.zip` file name at the top of the page. This will separate your request from previous requests made by your user.

An example for a UNIX interface via `sftp` is provided below:

```
$ sftp me@data.nyuwireless.com
me@data.nyuwireless.com's password:
Connected to data.nyuwireless.com.
sftp> ls
s1416856547827.zip        s1416856547827.zip
sftp> get s1416856547827.zip
Fetching /files/s1416856547827.zip to s1416856547827.zip
/files/s1416856547827.zip              63%   32MB  10.2MB/s   00:01 ETA
sftp> exit
$
```

2 Adding Users

Adding users is a two-step process that requires (1) adding users to the machine so that they can download data and (2) adding users to the permissions files so that they can access the website.

Please note: Passwords are currently 22 (or more) characters in length. Each character is one of–letter (lower or uppercase), number, or international symbol (above number keys). Ambiguous characters like l and I are artificially removed.

2.1 Adding Users to the Machine

To download data from the server to a local machine, the `sftp` protocol is used. Because of this, a unix user account needs to be created on the host (server) machine. This machine lives at `loki.poly.edu` or `data.nyuwireless.com`.

In order to ensure that these additional users do not get untoward access to the machine, their accounts are restricted access to a shell and they are chroot / jailed to a different directory. This process has been streamlined and two scripts: `AddUserScript.sh` and `DelUserScript.sh`. Both scripts live at `/web/tools/`.

To add a user to the system, it suffices to run `./AddUserScript.sh myNewUser` where `myNewUser` is the username of the account you wish to create. You must be root (or have sudo access) to run this script.

After adding the user, please be sure to verify that the account has been properly jailed and chrooted. This can be done by performing the following steps:

1. Attempt to ssh to the new user by doing

   `ssh myNewUser@loki.poly.edu`.

   The shell should return:
   This service allows sftp connections only. Connection to loki.poly.edu closed.

2. Verify jailing/chrooting worked – sftp to the machine by doing

   `sftp myNewUser@loki.poly.edu`

   and doing

   `$ pwd`.

This should return:
Remote working directory: /files.

2.2 Adding Users to the Website

All website files that need to be modified for permissions live in the `public` directory which can be found at the absolute path `/web/public/` on the host machine, `loki.poly.edu` or `data.nyuwireless.com`. Editing any of these files requires root or sudo access.

All `php` permissions are handled by the `SimpleAuth.php` script at `/web/public/`. At the top of this file is an array `$users` which stores usernames and their corresponding passwords.

```
<?php

class SimpleAuth
{
  // The users List (‘Login’ => ‘Password’)
  var $users = array(
    ‘user1’ => ‘abcdefghijklmnop’,
    ‘user2’ => ‘01234567890’,
    ‘user3’ => ‘password1234’
  );

  ...
```

To add a user, first edit this file by adding the new user / password to the end of the list. Make sure to append a comma to the last line first!

After editing the `SimpleAuth` script, you have to explicitly tell each of the files that access it to allow the added user. As of the date of this writing (27 April 2015), the files which rely on `SimpleAuth` are:

```
/web/public/docsI73.zip
/web/public/docsF73.zip
/web/public/docs73.zip
/web/public/send_form_email.php
/web/public/index.php
/web/public/docs28.zip
```

(To recover an updated list, it suffices to run `$ grep -r "SimpleAuth.php" .` when in `/web/public/` and parse the resulting list.)

At the top of each of these files, the `SimpleAuth` script is required and then a list of "valid" usernames is passed in.

```
${(require_once('SimpleAuth.php'))}->protectme(array(
    'allowedUsers' => array(
    'user1',
    'user2',
    'user3')
));
```

Upon adding a new user, the new username must be appended to this list (in each file).

After performing all of the above steps. It should be verified that the new user account works as desired.

This process involves:

1. Logging on to `loki`.
2. Attempting to view / download documentation.
3. Attempting to query the dataset.
4. Attempting to download data using the sftp protocol.

3 Adding Data

Data is meant to be added in chunks corresponding to *measurement campaigns*. The measurement and processing teams should provide you with three sets of data (1) raw data files, (2) processed data files, (3) processed statistics file(s), a data documentation file.

3.1 Adding Data to the Server

You will need to create a folder for all data corresponding to the new measurement campaign to reside. You will need root / sudo access to do this.

1. Make a folder in `/web/data/` with the name of your campaign (e.g. `28GHz`). For our purposes, I shall refer to this folder as `root/`.
2. Within `root/`, create subfolders: `PngDB/`, `ProcessedDB/`, `MetaDB/`, `Common/`, and `RawDB/`.
3. `chmod -r` the `root/` directory (this is a recursive call) to `777`.
4. Copy `AA_README.txt` from `/web/data/Common/` to `root/Common/`.
5. Upload all image files to `root/PngDB/`.
6. Upload data documentation file to `root/`.
7. Add all raw data files to `root/RawDB/`.
8. Zip all data files from `root/RawDB/` and move zip file to `root/`.
9. Add map to `/web/public/map***.png` where `***` corresponds to the name of your campaign, as specified in Step 1.
10. Create a file called `dynamic_img***.png` in `/web/public/cgi-bin/` and point it to your image directory. Use `/web/public/cgi-bin/dynamic_img.png` as a template, if necessary.
11. Copy the `*ParameterDatabase*` text file to `root/`.
12. Run the auto-loader python script from within `/web/tools/SUITE` by passing in the template file corresponding to the measurement campaign of interest. More detailed instructions are provided below.
13. After running `python AUTO.py`, add the name of the campaign to the `campaign_list` variable in the `query.cgi` file that lives in `/root/web/public/cgi-bin/`.

14. Finally, add the map for the measurement campaign to the `/root/web/public/` folder using the naming convention `map<campaign-name>.png`.

3.2 Appendix

3.2.1 What do the experimentalists give us?

When performing an measurement, the experimentalists record measured power versus time onto csv files. These data are written into a file that is linked with metadata (including location, calibration, and hardware information).

The post-processing procedure takes calibration data (obtained the same day as the real experiments) and adjusts raw measured data accordingly. The adjusted data is saved to new csv files and is outputted to image files as well. These newly produced files have names corresponding to transmitter azimuth/elevation, receiver azimuth/elevation and other such information.

The post-processing also produces the large csv file which we term the *"parameter database"* and contains metadata and summary statistics for each measurement in the campaign.

The overall dataflow is, thus:

calibration data, raw measurements → images, parameter database, processed data files

3.2.2 How do we store the experimentalists' data?

Often, `/web/data/<campaign-name>/RawDB` has subfolders corresponding to each date of measurement. These subfolders contain calibration data.

When a query is made, the user is effectively searching against the large csv parameter file and receives: data corresponding to the relevant row(s), images corresponding to these rows, and the original raw data files.

3.2.3 Data template file format

Here is an example of a way to fill in the `scaffold.template` file which the data providers will supply alongside their other contributions. This example corresponds to the 28 GHz indoor measurement campaign which took place during the summer of 2014.

1. For this measurement campaign, we name the file `28GHz_Indoor_2014.template`. While you are free to choose whatever filename you wish, we recommend explicit verbosity.

2. **parfile name** is the name of the parameter file which is supplied by the data provider. This is the Giant spreadsheet corresponding to every processed measurement. In this case, **parfile name** is `28GHzIndoorParameterDatabaseSummer2014.csv`.

3. **Format string** is the way we form filenames from the parfile one row at a time. In particular, we take one row from the parfile at a time and replace the % signs by the values in that row in order to match the filenames of the given image files.

   For example, the format string `%_RX%_%_Meas%_Rot%_RX_AZ_%_TX_AZ_%_PDB` would correspond to the filename `MTC4_RX13_TXsweep_Meas8_Rot3_RX_AZ_300_TX_AZ_30_PDP.png`.

4. In `CSV_PNG_TYPE`, we have to say which fields (columns) of a csv row should correspond to the percent signs, left to right. For example, a typical line in the csv file would be:

   MTC1,1,1,1,30,-5,2.5,1.5,2.80e+10,15,28.8,30,V,15,28.8,30,V,LOSNB,6.4,180,-8, 150,8, -57.3384, 91.0384,29.6535,4,3.6783,3.7,400000000,-82.0823,4,2.5,3.1761,2.5, 9.625,17.25, 07.18.14,1, IQsquared_MTC1_1_4.000000m_Tx(Az0.0)(EL0.0)_Rx(Az-150.0)(EL8.0)_1.txt, TXsweep

   and the corresponding `CSV_PNG_TYPE` would be `0,1,40,2,5,21,19`

After filling out the data, run `python AUTO.py 28GHz_Indoor_2014.template` from `/web/tools/SUITE`.

`AUTO.py` knows to go to the subdirectory `28GHz_Indoor_2014` of `/web/data` which is where the data provider uploaded the csv parameter file as well as the PNG image files.

Sample `template` file below:

```
# Enter the campaign name below.
# It should be formed like "28GHz"
# or "73_GHz_Indoor" ... Note that
# this string must be unique among
# all campaigns!
CAMPAIGN = "28GHz_Indoor_2014"

# Enter the name of the parfile
# which should be located in the
# root of the data directory.
PARFILE_NAME = "28GHzIndoorParameterDatabaseSummer2014.csv"

# Enter the format string for the
# png filenaming convention, substituting
# the percent sign '%' for changing
# values. In the processing script
# the percent signs will be replaced
# by values taken from the parameter
# file, as below.
# EXAMPLE:
# %_%_config%_step_%_azi%_el%_azi%_el%
CSV_PNG_FSTR = "%_RX%_%_Meas%_Rot%_RX_AZ_%_TX_AZ_%_PDP
# MTC4_RX13_TXsweep_Meas8_Rot3_RX_AZ_300_TX_AZ_30_PDP.pdf

# Enter the numbers of the columns
# in the csv parfile which correspond
# to the percent wildcards above.
# For example, if the first wildcard
# represents the TX_NAME, then enter
# the number corresponding to the column
# in the csv parfile which stores the
# name of the transmitter for a given
# measurement (line).
# EXAMPLE:
# "0,1,2,5,20,21,22,23"
CSV_PNG_TYPE = "0,1,40,2,5,21,19"
```